HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P. O. Box 272400
Fort Collins, Colorado 80527-2400

11-01-00

PATENT APPLICATION

ATTORNEY DOCKET NO. 10001275-1

A

IN THE U.S. PATENT AND TRADEMARK OFFICE
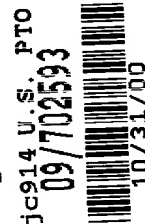### Patent Application Transmittal Letter

COMMISSIONER FOR PATENTS
Washington, D.C. 20231

Sir:

Transmitted herewith for filing under 37 CFR 1.53(b) is a(n): (X) Utility ( ) Design

(X) original patent application,

( ) continuation-in-part application

U.S. PTO
09/702593
10/31/00

INVENTOR(S): **Cary A. Coutant and Carol L. Thompson**

TITLE: **Method and Apparatus for Switching Between Multiple Implementations of a Routine**

Enclosed are:

(X) The Declaration and Power of Attorney. (X) signed ( ) unsigned or partially signed

(X) __3__ sheets of drawings (one set) ( ) Associate Power of Attorney

( ) Form PTO-1449 ( ) Information Disclosure Statement and Form PTO-1449

( ) Priority document(s) ( ) (Other) _____ (fee $_____)

(1) FOR	(2) NUMBER FILED	(3) NUMBER EXTRA	(4) RATE	(5) TOTALS
CLAIMS AS FILED BY OTHER THAN A SMALL ENTITY				
TOTAL CLAIMS	16 — 20	0	X $18	$ 0
INDEPENDENT CLAIMS	5 — 3	2	X $80	$ 160
ANY MULTIPLE DEPENDENT CLAIMS	0		$270	$ 0
BASIC FEE: Design ($320.00); Utility ($710.00)				$ 710
TOTAL FILING FEE				$ 870
OTHER FEES				$
TOTAL CHARGES TO DEPOSIT ACCOUNT				$ 870

Charge $ __870__ to Deposit Account 08-2025. At any time during the pendency of this application, please charge any fees required or credit any over payment to Deposit Account 08-2025 pursuant to 37 CFR 1.25. Additionally please charge any fees to Deposit Account 08-2025 under 37 CFR 1.16, 1.17,1.19, 1.20 and 1.21. A duplicate copy of this sheet is enclosed.

"Express Mail" label no. __EL571600960US__

Date of Deposit __Oct. 31, 2000__

I hereby certify that this is being deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under 37 CFR 1.10 on the date indicated above and is addressed to: Commissioner for Patents, Washington, D.C. 20231

By _____

Typed Name: Lynda J. Bauer

Respectfully submitted,

Cary A. Coutant and Carol L. Thompson

By _____

LeRoy D. Maunu

Attorney/Agent for Applicant(s)
Reg. No. 35,274

Date: Oct. 31, 2000

Telephone No.: (651) 686-6633

Rev 10/00 (TransNew)

- Attach as First Page to Transmitted Papers -

1   METHOD AND APPARATUS FOR SWITCHING BETWEEN MULTIPLE

2   IMPLEMENTATIONS OF A ROUTINE

3   Cary A. Coutant

4   Carol L. Thompson

5

FIELD OF THE INVENTION

7   The present invention generally relates to management of binary program code

8   for different implementations of a processor architecture, and more particularly to

9   switching between multiple implementations of a routine that are associated with the

10 implementations of a processor architecture.

11

BACKGROUND

13   It is common to build multiple implementations of a basic processor

14 architecture for the purpose of providing various performance and pricing options. For

15 example, a processor architecture that supports a particular instruction set may have a

16 first-level cache in one implementation, and another implementation may have first and

17 second level caches.

18   Some system-provided routines and other library routines are written and

19 compiled to exploit the performance characteristics of a particular implementation of a

20 processor. For example, a memory-to-memory copy routine may be written and

21 compiled differently from one implementation to another. While the binary code will

22 execute correctly on any implementation, there may be a negative impact on

23 performance when the binary code is executed on an implementation other than the

24 target implementation.

25   A software developer can either develop separate binary libraries for the

26 different implementations, develop a single binary for all implementations, or develop

27 several versions of selected routines along with a run-time switch for selecting between

28 the different versions. Developing separate binary libraries is technically

29 straightforward. However, there is a cost associated with managing and distributing

30 separate binary libraries. If only a single binary library is provided, users may not

31 receive the full performance benefit of a particular implementation. While run-time

32 switches would appear to provide a reasonable tradeoff between the management of

33 multiple binary libraries and performance degradation, the run-time switch introduces

34 overhead when a library call is made to determine the implementation on which the

1   code is executing.

2   A method and apparatus that address the aforementioned problems, as well as

3   other related problems, are therefore desirable.

4

5   SUMMARY OF THE INVENTION

6   In various embodiments, a method and apparatus are provided for switching

7   between multiple implementations of a routine. In one embodiment, different versions

8   of a library routine are programmed to exploit different features of different computer

9   systems. The different versions are available in a single library, and an application

10   program need not differentiate between the different implementations in using the

11   routine. Using hardware characteristics that are associated with the different versions

12   and hardware characteristics of the computer system on which the application is to be

13   executed, references to the routine are resolved when the application and library are

14   loaded. Thus, execution of the application is not burdened with runtime resolution of

15   references to the routine.

16   It will be appreciated that various other embodiments are set forth in the

17   Detailed Description and Claims which follow.

18

19   BRIEF DESCRIPTION OF THE DRAWINGS

20   Various aspects and advantages of the invention will become apparent upon

21   review of the following detailed description and upon reference to the drawings in

22   which:

23   FIG. 1 is a flowchart of a process for generating multiple binary

24   implementations of a routine for different implementations of a particular processor

25   architecture;

26   FIG. 2 is a block diagram illustrating a symbol table in relationship to a shared

27   library of object code modules; and

28   FIG. 3 is a flowchart of a process for loading a library routine in accordance

29   with one embodiment of the invention.

30

31   DETAILED DESCRIPTION

32   In various embodiments, the invention provides a technique for switching

33   between multiple implementations of a library routine that are available in a library of

34   routines. Each implementation of a routine has an associated set of hardware

1   characteristics that indicate the hardware on which the implementation is intended to

2   execute. The hardware characteristics may include, for example, the processor clock

3   speed or a model number, cache configuration, latency of selected hardware operations

4   (load and store, for example), and the availability of certain extensions to the

5   instruction set. When a routine having multiple implementations is loaded, the

6   reference is resolved to the appropriate implementation using the associated hardware

7   characteristics and the hardware characteristics of the host system. Additional

8   hardware characteristics that may be used for different implementations of routines

9   include, for example, bypass characteristics, branch prediction behavior, pre-fetching

10   capability, information describing stall conditions, branch penalties, size and

11   associativity of processor data structures (not just cache, but branch prediction and

12   ALAT-like structures as well), queue sizes for out-of-order or decoupled processors,

13   and the number of processors in a multi-processor system.

14   FIG. 1 is a flowchart of a process for generating multiple binary

15   implementations of a routine for different implementations of a particular processor

16   architecture, in accordance with one embodiment of the invention. The process

17   generally entails for each implementation of a routine, generating object code modules

18   for the different implementations and adding entries in a symbol table for the different

19   object code modules. An entry in the symbol table for a routine having multiple

20   implementations has an associated set of hardware characteristics. The hardware

21   characteristics are those of the platform on which the associated object code module is

22   intended to execute.

23   At step 102, the first (or next, depending on the iteration) implementation of the

24   routine is obtained for processing, and at step 104 the hardware characteristics

25   associated with the routine are obtained.

26   The symbol table is updated at step 106. The symbol table includes names of

27   routines and references to the associated object code modules. For routines having

28   multiple implementations, hardware characteristics are also associated with the routine

29   name in the symbol table. When an application is loaded and bound to the shared

30   library that contains the multiple binary implementations of a routine, the system

31   dynamic loader selects the appropriate implementation from the symbol table based on

32   the hardware characteristics of the host system.

33   In one embodiment, the hardware characteristics that are to be associated with a

34   routine are provided by the developer in a configuration file that is read by the linker at

1   the time the shared library is being built. In this embodiment, the programmer will

2   have coded different versions of the routine and used different names for the different

3   versions. The information in the configuration file associates the names of the different

4   versions with sets of hardware characteristics and with a generic name.

5   In another embodiment, the compiler could be adapted to generate multiple

6   object code modules from a source code module. For example, in response to a

7   command-line option, the compiler automatically generates hardware-specialized

8   implementations, generates unique symbol names for the specialized implementations,

9   and generate the mapping information. The mapping information associates the generic

10 routine name with the specialized implementations, and the specialized

11 implementations with sets of hardware characteristics. The mapping information may

12 be stored either in the object file or in a separate configuration file that is used by the

13 linker, for example.

14   At step 108, the object code module is created for the routine, and the object

15 code module is added to the shared library at step 110. Each routine in the shared

16 library is assigned a unique name.

17   At step 112, the entry in the symbol table (step 106) is updated to reference the

18 associated object code module in the object code library. Decision step 114 tests

19 whether there are additional implementations to process. If so, control is returned to

20 step 102. Otherwise, the process for generating the multiple implementations is

21 complete.

22

23   FIG. 2 is a block diagram illustrating a symbol table in relationship to a shared

24 library of object code modules. The routines in shared library 152 are object code

25 modules that are associated with and referenced by the entries in symbol table 154.

26   Routines 1 - (n+1) are illustrated in shared library 152. Routines 1 - n have

27 single implementations, and two implementations are illustrated for example routine (n

28 + 1). The first implementation of routine (n + 1) is named routine (n + 1), and the

29 second implementation of routine (n + 1) is named routine (n + 1)'.

30   Symbol table 154 includes entries for each routine and implementation.

31 Routines 1 - n, having only single implementations, include only the routine name and

32 a reference to the corresponding object code module in shared library 152. Routine (n

33 + 1) has two implementations, and the entries associated therewith include respective

34 sets of hardware characteristics that describe, for example, the processor for which the

4

1   implementations were developed. The entry having hardware characteristics set 1

2   references routine (n + 1) code in shared library 152, and the entry having hardware

3   characteristics set 2 references routine (n + 1)' code in the shared library.

4   When an application is loaded and bound to shared library 152, the system

5   dynamic loader selects the appropriate binary implementation from the symbol table

6   based on the hardware characteristics of the host processor. Thus, the reference to the

7   appropriate binary implementation is resolved when the program using the shared

8   library is loaded. Alternatively, some environments may load the shared library after a

9   program begins execution, and in this environment the references are resolved when the

10   shared library is loaded. In either environment, the references are resolved at load time

11   versus runtime. The resolution of the references to routines in the symbol table results

12   in code references to the addresses of the binary implementations in the shared library

13   152.

14   By selecting the appropriate object code routine once when the routine is first

15   referenced instead of resolving the reference each time the routine is referenced at run-

16   time, the overhead for the switch occurs in the compiler and loader, thereby eliminating

17   issues with respect to run-time performance and switching to an appropriate

18   implementation of a routine.

19

20   FIG. 3 is a flowchart of a process for loading a library routine in accordance

21   with one embodiment of the invention. At step 302, the hardware characteristics are

22   obtained from a system configuration file, for example. In another embodiment, the

23   characteristics may be obtained, for example, from hardware identification registers or

24   from firmware.

25   At step 304, the loader obtains the name of the routine to be loaded. For

26   example, an application program may reference a particular shared library routine, and

27   the loader uses the program-specified routine name to locate the proper object code

28   module in the shared library.

29   The routine name and hardware characteristics are used at step 306 to match an

30   entry in symbol table 154. Using the reference in the matching entry, step 208 loads

31   the referenced object that is associated with the hardware characteristics. Forward from

32   the time that a routine is referenced and the proper object code module is identified and

33   loaded, no further matching of hardware characteristics is required on subsequent

34   references to the routine.

1   The present invention is believed to be applicable to a variety of systems that

2   switch between multiple implementations of a routine based on hardware

3   characteristics. Other aspects and embodiments of the present invention will be

4   apparent to those skilled in the art from consideration of the specification and practice

5   of the invention disclosed herein. It is intended that the specification and illustrated

6   embodiments be considered as examples only, with a true scope and spirit of the

7   invention being indicated by the following claims.

8

1   **CLAIMS**

2   What is claimed is:

3   1.   A computer-implemented method for switching between multiple

4   implementations of a routine in a library of routines that are linked with an application

5   program that is hosted by a computer system, comprising:

6   compiling a plurality of implementations of a routine into respective object code

7   modules, the routine having an associated name and each implementation adapted to a

8   selected hardware configuration;

9   associating the object code modules with the name of the routine and respective

10 sets of hardware characteristics; and

11   resolving when the application program is loaded into memory of the computer

12 system, a reference to the routine using the sets of hardware characteristics and a

13 hardware configuration of the system.

14

15 2.   The method of claim 1, further comprising establishing a symbol table having a

16 plurality of entries, each entry including a name of a routine and a reference to an

17 object code module in the library.

18

19 3.   The method of claim 2, further comprising, for the routine having a plurality of

20 implementations, adding a plurality of entries to the symbol table and associating

21 respective sets of hardware characteristics with the plurality of entries.

22

23 4.   The method of claim 3, wherein the hardware characteristics include at least one

24 of clock speed of the processor, processor model, cache configuration of the system,

25 hardware operation latency times, instruction set characteristics, bypass characteristics,

26 branch prediction behavior, pre-fetching capability, information describing stall

27 conditions, branch penalties, size and associativity of processor data structures, queue

28 sizes for out-of-order or decoupled processors, and the number of processors in a multi-

29 processor system.

30

31 5.   The method of claim 4, wherein the resolving step further comprises obtaining

32 the hardware configuration of the system from at least one of a system configuration

33 data file, one or more system identification registers, and system firmware.

34

1   6.   The method of claim 3, wherein the resolving step further comprises obtaining

2   the hardware configuration of the system from at least one of a system configuration

3   data file, one or more system identification registers, and system firmware.

4

5   7.   The method of claim 1, wherein the hardware characteristics include at least one

6   of clock speed of the processor, processor model, cache configuration of the system,

7   hardware operation latency times, and instruction set characteristics.

8

9   8.   The method of claim 1, wherein the resolving step further comprises obtaining

10   the hardware configuration of the system from at least one of a system configuration

11   data file, one or more system identification registers, and system firmware.

12

13   9.   A computer-implemented method for switching between multiple

14   implementations of a routine in a library of routines that are linked with an application

15   program hosted by a computer system, comprising:

16   establishing a set of hardware configuration characteristics that describe the

17   computer system;

18   establishing a symbol table, the symbol table having one or more entries that

19   include a name of a routine, a set of hardware characteristics, and an address

20   referencing a routine in the library;

21   obtaining a name of a routine having multiple implementations when the library

22   is loaded with the application program into memory of the computer system;

23   matching the name of the routine and the set of hardware configuration

24   characteristics that describe the computer system to an entry in the symbol table; and

25   generating an address in executable code for references to the routine having

26   multiple implementations when the library is loaded with the application program, the

27   address referencing an implementation in the library as identified in the matching step

28   by the entry in the symbol table.

29

30   10.   The method of claim 9, wherein the hardware configuration characteristics

31   include at least one of clock speed of the processor, processor model, cache

32   configuration of the system, hardware operation latency times, and instruction set

33   characteristics.

34

1   11.   The method of claim 10, wherein the resolving step further comprises obtaining

2   the hardware configuration of the system from at least one of a system configuration

3   data file, one or more system identification registers, and system firmware.

4

5   12.   The method of claim 9, wherein the resolving step further comprises obtaining

6   the hardware configuration of the system from at least one of a system configuration

7   data file, one or more system identification registers, and system firmware.

8

9   13.   An apparatus for switching between multiple implementations of a routine in a

10   library of routines that are linked with an application program that is hosted by a

11   computer system, comprising:

12   means for compiling a plurality of implementations of a routine into respective

13   object code modules, the routine having an associated name and each implementation

14   adapted to a selected hardware configuration;

15   means for associating the object code modules with the name of the routine and

16   respective sets of hardware characteristics; and

17   means for resolving when the application program is loaded into memory of the

18   computer system, a reference to the routine using the sets of hardware characteristics

19   and a hardware configuration of the system.

20

21   14.   A computer-implemented symbol table for referencing a library of object code

22   modules that implement a plurality of routines, comprising:

23   a first set of one or more entries, each entry in the first set including a unique

24   name of a routine and a reference to an object code module in the library; and

25   a second set of one or more entries, each entry in the second set including a

26   shared name of a routine, a set of hardware characteristics, and a reference to an object

27   code module in the library.

28

29   15.   The symbol table of claim 14, wherein the hardware characteristics include at

30   least one of clock speed of a processor, processor model, cache configuration, hardware

31   operation latency times, instruction set characteristics, bypass characteristics, branch

32   prediction behavior, pre-fetching capability, information describing stall conditions,

33   branch penalties, size and associativity of processor data structures, queue sizes for out-

9

1   of-order or decoupled processors, and the number of processors in a multi-processor

2   system.

3

4   16.   A computer program product configured for causing a computer to perform the

5   steps of:

6   compiling a plurality of implementations of a routine into respective object code

7   modules, the routine having an associated name and each implementation adapted to a

8   selected hardware configuration;

9   associating the object code modules with the name of the routine and respective

10   sets of hardware characteristics; and

11   resolving when the application program is loaded into memory of the computer

12   system, a reference to the routine using the sets of hardware characteristics and a

13   hardware configuration of the system.

14

15

1   **ABSTRACT**

2   Method and apparatus for switching between multiple implementations of a

3   routine. A plurality of implementations of a routine are compiled into respective object

4   code modules. In one embodiment, each implementation of the routine is adapted for a

5   particular hardware configuration. The different object code modules are associated

6   with respective sets of hardware characteristics and with the name of the routine.

7   When the application program and library are loaded into memory of the computer

8   system, a references to the routine are resolved using the sets of hardware

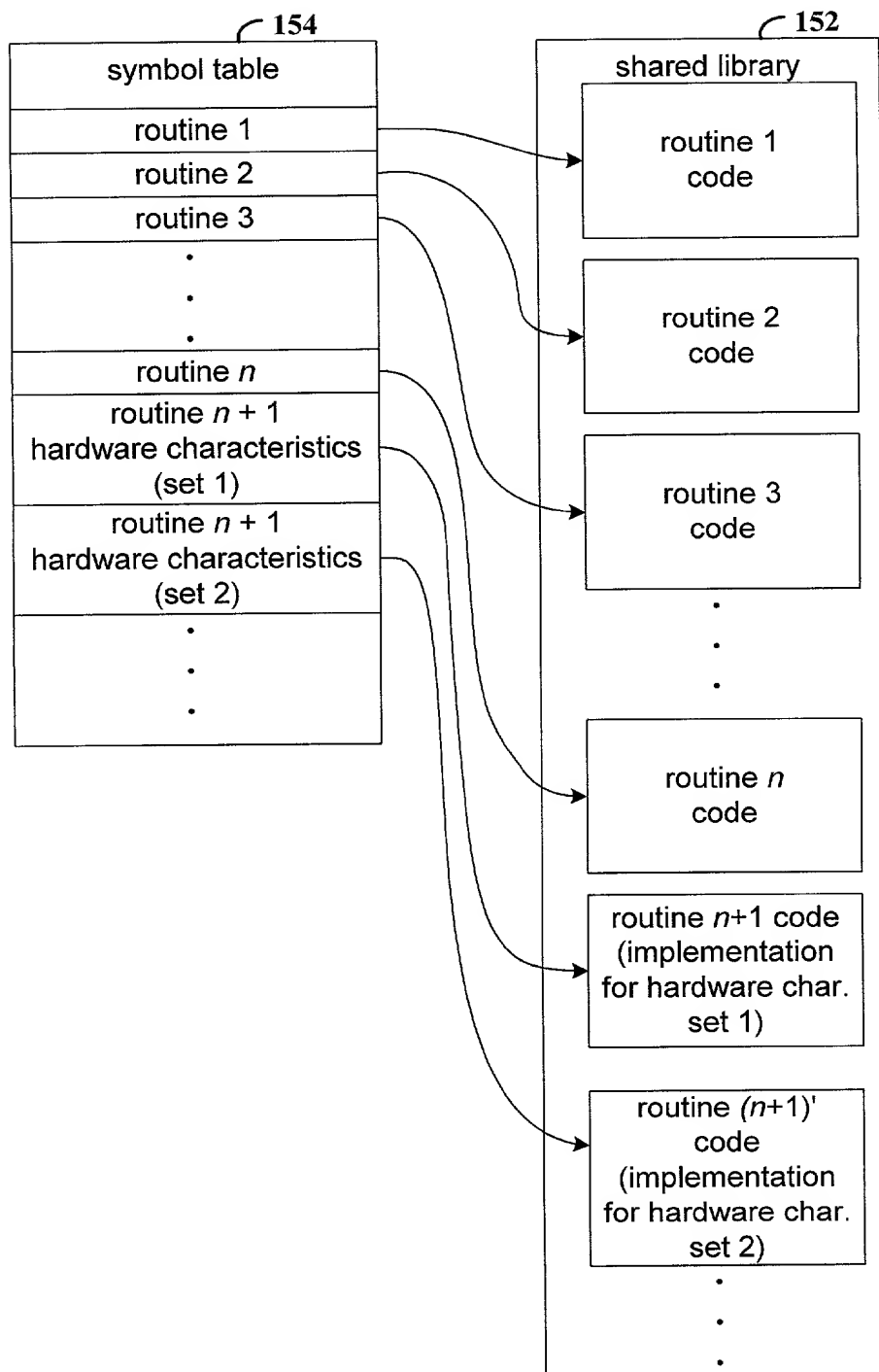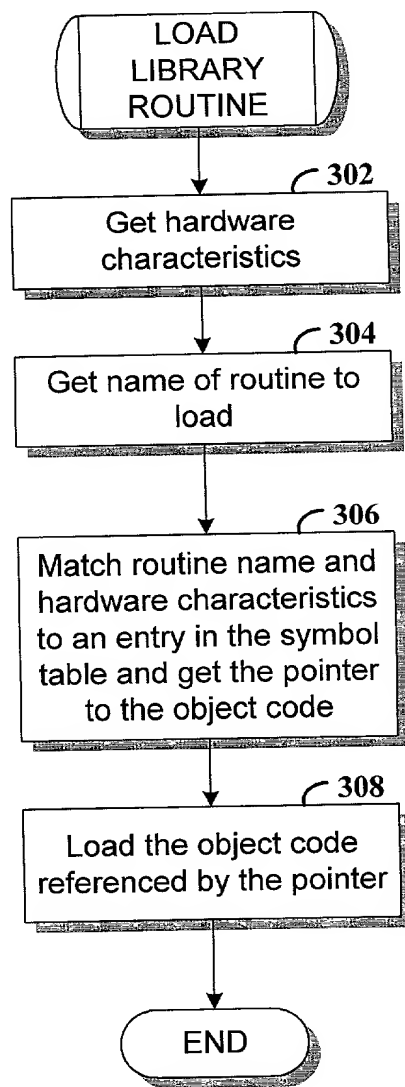9   characteristics and the hardware configuration of the system.

GENERATION OF
BINARY CODE FOR
MULTIPLE
IMPLEMENTATIONS
OF A ROUTINE

114

More
implementations?

—Yes—

No

102

Get the first/next
implementation of the
routine

END

104

Get the hardware
characteristics associated
with the implementation

106

Add entry to the symbol
table for the
implementation, including
the name of the
implementation and the
associated hardware
characteristics

108

Generate object code for
the implementation

110

Add the object code to a
shared library using a
unique name

112

Link the entry in the
symbol table to the object
code in the shared library

FIG. 1

FIG. 2

FIG. 3

DECLARATION AND POWER OF ATTORNEY FOR PATENT APPLICATION	ATTORNEY DOCKET NO. 1000-1275

As a below named inventor, I hereby declare that:

My residence/post office address and citizenship are as stated below next to my name;

I believe I am the original, first and sole inventor (if only one name is listed below) or an original, first and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled:

Method and Apparatus for Switching between Multiple Implementations of a Routine

the specification of which is attached hereto unless the following box is checked:

( ) was filed on _____ as US Application Serial No. or PCT International Application Number _____ and was amended on _____ (if applicable).

I hereby state that I have reviewed and understood the contents of the above-identified specification, including the claims, as amended by any amendment(s) referred to above. I acknowledge the duty to disclose all information which is material to patentability as defined in 37 CFR 1.56.

Foreign Application(s) and/or Claim of Foreign Priority

I hereby claim foreign priority benefits under Title 35, United States Code Section 119 of any foreign application(s) for patent or inventor(s) certificate listed below and have also identified below any foreign application for patent or inventor(s) certificate having a filing date before that of the application on which priority is claimed:

COUNTRY	APPLICATION NUMBER	DATE FILED	PRIORITY CLAIMED UNDER 35 U.S.C. 119	
			YES: ___	NO: X
			YES: ___	NO: X

Provisional Application

I hereby claim the benefit under Title 35, United States Code Section 119(e) of any United States provisional application(s) listed below:

APPLICATION SERIAL NUMBER	FILING DATE

U. S. Priority Claim

I hereby claim the benefit under Title 35, United States Code, Section 120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code Section 112, I acknowledge the duty to disclose material information as defined in Title 37, Code of Federal Regulations, Section 1.56(a) which occurred between the filing date of the prior application and the national or PCT international filing date of this application:

APPLICATION SERIAL NUMBER	FILING DATE	STATUS (patented/pending/abandoned)

POWER OF ATTORNEY:

As a named inventor, I hereby appoint the following attorney(s) and/or agent(s) to prosecute this application and transact all business in the Patent and Trademark Office connected therewith:

Customer Number | 022879

Place Customer Number Bar Code Label here

Send Correspondence to:
HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P.O. Box 272400
Fort Collins, Colorado 80527-2400

Direct Telephone Calls To:

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Full Name of Inventor: **Cary A. Coutant**   Citizenship: **USA**

Residence: **13478 Briar Court, Saratoga, California 95070**

Post Office Address: **13478 Briar Court, Saratoga, California 95070**

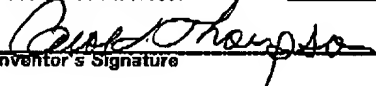Inventor's Signature _____   Date **10/27/00**

DECLARATION AND POWER OF ATTORNEY
FOR PATENT APPLICATION (continued)

ATTORNEY DOCKET NO. **1000-1275**

Full Name of # 2 joint inventor: **Carol L. Thompson**   Citizenship: **USA**

Residence: **6937 Calabazas Creek Circle, San Jose, California 95129**

Post Office Address: **6937 Calabazas Creek Circle, San Jose, California 95129**

Inventor's Signature   Date **10/27/00**

Full Name of # 3 joint inventor: _____   Citizenship: _____

Residence: _____

Post Office Address: _____

Inventor's Signature _____   Date _____

Full Name of # 4 joint inventor: _____   Citizenship: _____

Residence: _____

Post Office Address: _____

Inventor's Signature _____   Date _____

Full Name of # 5 joint inventor: _____   Citizenship: _____

Residence: _____

Post Office Address: _____

Inventor's Signature _____   Date _____

Full Name of # 6 joint inventor: _____   Citizenship: _____

Residence: _____

Post Office Address: _____

Inventor's Signature _____   Date _____

Full Name of # 7 joint inventor: _____   Citizenship: _____

Residence: _____

Post Office Address: _____

Inventor's Signature _____   Date _____

Full Name of # 8 joint inventor: _____   Citizenship: _____

Residence: _____

Post Office Address: _____

Inventor's Signature _____   Date _____

Rev 10/00 (DecPwr)   (Use Page Two For Additional Inventor(s) Signature(s))   Page 2 of 2